一种在矩阵空间中识别文本蕴涵的动态交互网络 *

霍 欢[1,2]，刘 亮[1†]

(1. 上海理工大学 光电信息与计算机工程学院，上海 200093; 2. 复旦大学 上海市数据科学重点实验室，上海 201203)

摘 要：针对文本蕴涵问题提出一种动态交互网络（dynamic interactive network，DIN）进行识别。不同于已有交互模型，DIN 将两句词向量投射到二维矩阵空间中进行交互，然后利用输出矩阵为同时处理上下文信息和控制信息流动的 GRU 编码器生成动态权重。前者通过更高阶形式的信息交互挖掘深层逻辑片段，后者通过改变交互信息与上下文信息的结合模式帮助编码器有效区分两者的重要性差异。模型在 SNLI 测试集上获得了 88.0%的识别准确度，超过已有最佳模型，且使用的训练参数仅为它的一半。

关键词：文本蕴涵识别；交互网络；矩阵空间；动态权重

中图分类号：TP399　　doi: 10.3969/j.issn.1001-3695.2018.03.0196

Dynamic interactive network over matrix-space for recognizing textual entailment

Huo Huan[1,2], Liu Liang[1†]

(1. *School of Optical-Electrical & Computer Engineering*, *University of Shanghai for Science & technology*, *Shanghai* 200093, *China*; 2. *Shanghai Key Laboratory of Data Science*, *Fudan University*, *Shanghai* 201203, *China*)

Abstract: This paper presented a dynamic interactive network (DIN) for recognizing textual entailment. Unlike the other interactive models, DIN facilitates the interaction by projecting the embedding vectors into a two-dimensional matrix space, and then uses the output matrices to produce dynamic weights for the GRU encoder that both processes the context information and controls the information flow. It empowers the extraction of logic segments through higher-orders of information interactions and helps the encoder better choose between the context and the interactive information. Experiments on the SNLI corpus show that our model achieves a test accuracy of 88.0%, outperforming the state-of-the-art with only a small amount of the training parameters introduced.

Key words: textual entailment recognition; interactive network; matrix space; dynamic weights

0 引言

随着自然语言处理领域研究的不断深入，如何让机器真正地理解自然语言，而不仅仅停留在对表层语义的理解上，成为了许多学者面临的问题[1]。围绕这一问题展开的众多研究中有一项基础性工作——识别文本蕴涵（recognizing textual entailment， RTE），它依托自然语言句子，旨在通过推理来识别前提句（premise，简称 P 句）和假设句（hypothesis，简称 H 句）之间存在的逻辑关系（下一节将给出详细定义），是对深层语义挖掘的重要实践，在语义搜索，人机对话，问答系统等领域都有重要的促进作用。

近年来，得益于神经网络在机器翻译研究上的成功，为识别文本蕴涵提供了诸多新的思路[2]，原本被广泛使用的传统识别模型由于繁复的推理流程逐渐被弃用，可端到端训练的神经网络模型取而代之成为主流。一开始，这类模型普遍采用编码器对两句进行独立编码，常用的有两类：基于线性结构的（如 GRU[3]）和基于树结构的（如 quasi-TreeLSTM[4]）。在获得两个高度概括了上下文信息的编码向量后，通过一系列比较方法（如向量作差，元素相乘）将两者合二为一并输入分类器进行标签预测。但是，当一个句子经编码压缩后仅剩一个向量表示时，必定会忽略很多重要信息，导致此类模型的识别准确度很难有提升。

为此，模型[5,6]进一步提出利用关注力机制[7]来捕捉词与词间的语义联系。相较之前模型，关注力机制通过逐词匹配进行推理，充分利用原本被忽视的保存在各词编码向量内的信息，大幅提升了原有模型的识别准确度。但是，由于编码完后句子的信息就被固定在单个向量里，此类模型的交互性并不能体现在两句信息的流动上，因此仅被认为存在一定的弱交互性。

收稿日期：2018-03-15；**修回日期：**2018-05-15　**基金项目：**国家自然科学基金资助项目（61003031）；上海重点科技攻关项目（14511107902）；上海市工程中心建设项目（GCZX14014）；上海市一流学科建设项目（XTKX2012）；沪江基金研究基地专项资助项目（C14001）

作者简介：霍欢（1979-），女，副教授，博士，主要研究方向为云计算、数据挖掘及不确定数据流技术；刘亮（1991-），男（通信作者），硕士，主要研究方向为机器学习及自然语言处理（1904219831@qq.com）.

为了促进句子对间信息的真实流动，以便更好地建模句子对间的语义逻辑关系，一系列强交互模型被相继提出[7~10]，它们的一大特点是，在编码一个句子的过程中必须兼顾另一个句子已有的编码信息。由于强调了信息流动的重要性，强交互模型的识别准确度普遍高于独立编码和弱交互模型。

基于强交互的思想，本文提出一种高效的动态交互网络（DIN）来识别文本蕴涵。图 1 展示了 DIN 的结构全貌，它整体上遵循[5,6,7]的逐词交互方式。相较于已有强交互模型，DIN 做出了如下两点改变：

a)受连续语义（continuation semantics）[11]的启发，DIN 将两句中参与交互的词向量投射到二维矩阵空间进行交互。相较于传统交互模型中向量间的按元素乘法或加权和，矩阵乘法代表更高维信息间的交互，利于模型挖掘出隐藏在更深处的逻辑关系片段。

b) 受动态网络权重生成模型[12]的启发，在编码 H 句的某时刻t，利用交互获得的输出矩阵可以生成一组只针对当前时刻的权重三元组(W_r^t, W_z^t, W_h^t)，分别作为 GRU 编码器在此时刻的重置门（reset gate）,更新门（update gate）和候选激励（candidate activation）的计算权重。

值得注意的是，此处的 GRU 编码器在强交互模型中扮演两个角色：编码当前句的上下文信息和控制交互信息在两句间的流动。传统交互模型在逐词编码的过程中会将两句的交互向量作为一个额外的输入引入计算步骤，这种做法的一个弊端是容易混淆上下文信息和交互信息。由于两者同为输入，且都是向量形式，孰轻孰重对编码器来说很难权衡，并且当其中的一个包含过多的无效信息，很容易会覆盖另一个中的有效信息。相反地，DIN 利用上一步矩阵化带来的优势，将交互信息转换成编码器的计算权重，以和上下文信息不同的形式将两者融合，尽量减小它们因直接叠加带来的不利影响。

在 SNLI（Stanford natural language inference）数据集[13]上的实验证明 DIN 通过使用较少的训练参数获得了比已有交互模型更高的识别准确度。

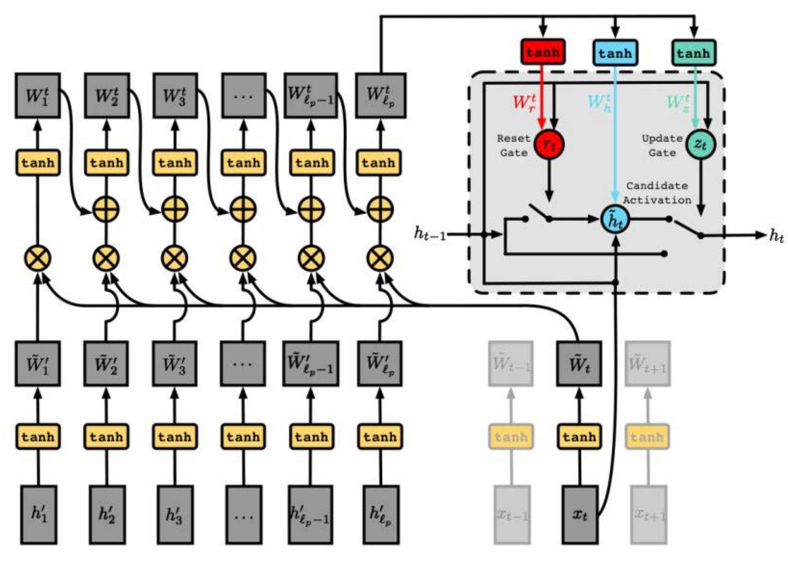


图 1 DIN 模型概览

1　相关工作

现有的为句子对建模的强交互模型并不多，此类模型强调在编码一个句子的同时需要兼顾另一个句子的信息。按照信息流动的方向，大致可以分成两类：

a)单向的（一般由 P 句向 H 句流动）。Lei 等人[7]提出的 rLSTM，对采用关注力机制进行逐词对齐的经典模型 word-by-word attention[5]进行修改，用编码 H 句过程中 LSTM 编码器的细胞状态（cell state）而非隐含状态（hidden state）同 P 句进行对齐操作，这意味着 P 句信息会直接对 H 句在各时刻的编码过程产生影响，输出的编码向量必定会包含彼此的交互信息。Liu 等人[8]提出的 SIN 抛弃关注力机制而直接改用两个控制门来匹配当前词与另一句各词。接着，增加一个额外的 LSTM cell 重读一遍当前词，过程中融入已有的交互信息。然而，完整的 SIN 模型包含以 P 句和 H 句为基准的两次方向相反的独立交互，涉及至少六次编码操作，训练成本非常高，很难用于实践。受 memory networks[14,15]的启发，Cheng 等人[16]提出的 LSTMN 在传统的 LSTM cell 里加入两块额外的内存区域，分别保存编码过程中各时刻的 hidden states 和 cell states，采用关注力机制来显式选择并读取它们，这在一定程度上缓解了传统 RNNs/LSTMs 中一直存在的长期依赖问题（long-term dependency）。进一步，为了适应句子对建模而设计的 deep fusion attention 模型，对两句都采用 LSTMN cell 编码，但在编码 H 句的各时刻会采用关注力机制与 P 句两个额外内存分别匹配，获得的交互信息会参与更新当前细胞状态。

b)双向的。文献[8,9]先后提出过两种强交互模型，分别是 DF-LSTMs 和 Coupled-LSTMs。不同于单向模型，它们对两句采取同步编码的方式，交互信息会同时参与对两句的编码过程。同 LSTMN 类似，DF-LSTMs 采用一块额外的内存区域来存储通过当前词与已有交互信息的比较获得新的交互信息。其中，已有的交互信息是通过关注力机制同步读取两句在各自外存内保存的交互信息组合而成的。与 SIN 类似，Coupled-LSTMs 也

抛弃关注力机制，通过网格化排布的 LSTM cells 来直接编码两句的上下文信息，因此获得的编码后的向量具有更强的耦合特性。

可以发现，上述模型的一个共同特点是为了满足信息流动性，会对编码器的计算过程进行一定的修改，将另一句编码信息融入到当前句的编码过程中。这种方法类似于经典的神经网络翻译模型[17]，缺点在于影响是间接产生的，无法清楚知道编码器自身捕获的上下文信息和额外的交互信息间的重要程度差异。相较而言，本文提出的动态交互模型 DIN 由动态生成的权重来承载信息的流动，影响是直接产生的，且可以通过变化的权重量化展示上面提及的差异性（参考图 3 和相关分析），利于人们对模型作用机理的理解。

2　DIN 模型

2.1　问题描述

识别文本蕴涵是机器理解自然语言的关键工作，核心是对自然逻辑（natural logics）[18]的理解。根据 MacCartney 等人[19,20]的叙述，一般认为存在着 16 种基本的语义逻辑关系，其中有 9 种是退化的，即它们的表达相对空洞，在实践中很难见到，剩下的 7 种逻辑关系被统一划分为三大类：蕴涵（entailment）、矛盾（contradiction）和中立（neutral）。表 1 列出的三个例子，它们拥有相同的 P 句：families waiting in line at an amusement park for their turn to ride the carousel（好多家庭在游乐场里排队坐旋转木马），根据 H 句的表述不同，被分别标记了不同的标签。第一个例子中，H 句中的 people 和 P 句中的 families 有明显的等价关系，at an amusement park 又可以找到完全对应的词句，因此两句被判定为存在蕴涵关系。第二个例子中，H 句中的 see a movie 和 P 句中 ride the carousel 两个动作存在不对称性，因此被判定为矛盾。第三个例子中，H 句中对餐厅的评价与 P 句没有任何逻辑上的关联性，因此无法给出明确的判定标签。这里的三个例子相对简单，机器要正确做出正确的判断也相对容易，但当某个句子存在过多冗余信息或者线索分布在多个不同位置需要整体把握时，机器就很难进行正确的推理了。

表 1　三个被不同标记但有相同前提句（Premise）的例子

假设句（Hypothesis）	类别
People are at an amusement park.　（人们在游乐场）	蕴涵
people are waiting to see a movie.　（人们在等待看电影）	矛盾
the restaurant is very bad.　（这家餐厅不怎么样）	中立

2.2　嵌入层

一个 RTE 任务可以由如下三元组表示：

$$(\boldsymbol{P},\boldsymbol{H},\boldsymbol{y}) \quad (1)$$

其中:$\mathrm{P}=\left(x_1',\cdots,x_{\ell_P}'\right)$代表长度为$\ell_P$的 P 句，$\mathrm{H}=\left(x_1,\cdots,x_{\ell_H}\right)$代表长度为$\ell_H$的 H 句，$y$是人工标注的标签（golden label）。在给定 P 和 H 情况下，模型要对两句的标签进行预测：

$$y^*=\mathrm{argmax}_{y\epsilon\gamma}\Pr(y|\boldsymbol{P},\boldsymbol{H}) \quad (2)$$

其中：$\gamma=\{\mathrm{Entailment},\mathrm{Contradiction},\mathrm{Neutral}\}$。条件概率$\Pr(y|P,H)$是一向量，每一维代表选择$\gamma$中一个标签的概率，$y^*$即是概率最大的那个标签，之后会进一步讨论它的求解。

在这里，每个x都是一个固定长度的词向量，一般使用预训练的词向量如 GloVe[21]或 word2vec[22]进行初始化。对超出词表范围的词则通过相同长度的随机向量进行初始化。

2.3　编码层

由图 1 可知，本文模型中信息是由 P 句向 H 句单向流动的，这意味着 P 句信息始终保持固定，为了更好地获得它的上下文信息，本层使用一个 GRU 编码器对它进行编码。GRU（Gated Recurrent Unit）最先由 Chung J 等[3]提出，它作为 LSTM（Long Short-Term Memory Networks）一种变体，摒弃了原本在 LSTM 中独立存在的记忆单元（memory cell），因此拥有更简单的控制门结构。

具体地，当前词$x_\tau',\tau\in[1,\cdots,\ell_P]$的编码向量$h_\tau'$是由此时刻的候选状态$\tilde{h}_\tau'$和上一时刻的状态$h_{\tau-1}'$通过线性差值函数获得：

$$\boldsymbol{h}_\tau'=(1-\boldsymbol{z}_\tau')\boldsymbol{h}_{\tau-1}'+\boldsymbol{z}_\tau'\widetilde{\boldsymbol{h}}_\tau' \quad (3)$$

其中:z_τ'是更新门（update gate），它决定了哪些信息需要被更新，具体计算过程如下：

$$\boldsymbol{z}_\tau'=\sigma(\boldsymbol{W}_z'\cdot[\boldsymbol{h}_{\tau-1}',\boldsymbol{x}_\tau']+\boldsymbol{b}_z') \quad (4)$$

可见，更新门是对已有状态$h_{\tau-1}'$和当前词x_τ'的线性求和。另一方面，候选状态$\tilde{h}_\tau'$的计算则与传统的循环单元类似：

$$\tilde{h}_\tau'=\tanh(\boldsymbol{W}_h'\cdot[\boldsymbol{r}_\tau'*\boldsymbol{h}_{\tau-1}',\boldsymbol{x}_\tau']+\boldsymbol{b}_h') \quad (5)$$

其中:r_τ'是重置门（reset gate），它决定了哪些信息需要被重置，具体与更新门类似：

$$\boldsymbol{r}_\tau'=\sigma(\boldsymbol{W}_r'\cdot[\boldsymbol{h}_{t-1}',\boldsymbol{x}_t']+\boldsymbol{b}_r') \quad (6)$$

上述的W_*'和b_*'是各个控制门计算的权重矩阵和偏移向量。

2.4　变换层

受编程语言理论（programming language theory）[20]的启发，Baker 等人[11]提出了连续语义（continuation semantics）的概念。他们认为，在自然语言中有很大一部分的表达(expression)是通过语义组合（semantic composition）获得的，但并非是简单的词与词之间的拼接，而是需要转换成更高阶的形式。本工作未直接涉及语义组合，但在某种程度上，词对间逻辑关系的匹配也类似于语义组合，因为两者的工作方式几乎如出一辙，都根据词对间的语义关系进行比较融合。由此，本层将 P 句的嵌入层词向量和 H 句经上一层编码后的词向量一起投射到二维矩阵空间，具体过程如下：

$$\widetilde{\boldsymbol{W}}=\tanh(\boldsymbol{W}_{trans}\boldsymbol{c}+\boldsymbol{b}_{trans}) \quad (7)$$

其中：$W_{trans}\epsilon\mathbb{R}^{\sqrt{d}\times\sqrt{d}\times d_c}$，$b_{trans}\epsilon\mathbb{R}^{\sqrt{d}\times\sqrt{d}}$，$d$是编码向量长度，$d_c$是向量$c$的长度。对 P 句，$c$是编码向量$h_t'\epsilon\mathbb{R}^d,\tau\in[1,\cdots,\ell_P]$；对 H 句，$\vec{c}$是词嵌入向量$x_t\epsilon\mathbb{R}^{d_{emb}},t\in[1,\cdots,\ell_H]$。变换层的输出$\widetilde{W}$是一个大小为$\sqrt{d}\times\sqrt{d}$的矩阵，考虑到根号的存在，在实践中$d$一般取 256、324 和 400 等可被开方的值。

2.5　交互层

经过上一层的转换，我们获得了各词比向量更高阶的矩阵表示。为了促进矩阵间的交互，本层采用和关注力机制类似的

逐词匹配方式，不同点在于，H 句当前词与 P 句各词进行对齐时，不会给 P 句各词分配权重，而是以更简单的矩阵乘法实现当前词与 P 句各词的逐一交互。具体地，H 句当前词x_t与 P 句某词x'_τ的交互方式如下：

$$\boldsymbol{W}_\tau^t = \tanh(\widetilde{\boldsymbol{W}}'_\tau \widetilde{\boldsymbol{W}}_t + \boldsymbol{W}_{inter}\boldsymbol{W}_{\tau-1}^t + \boldsymbol{b}_{inter}) \quad (8)$$

其中：$W_{inter} \in \mathbb{R}^{\sqrt{d}\times\sqrt{d}}$，$b_{inter} \in \mathbb{R}^{\sqrt{d}\times\sqrt{d}}$。实际上，上述计算过程可被类比为一个针对矩阵运算的简单的循环神经网络单元：$W_\tau^t = \mathrm{RNN}(W_{\tau-1}^t, \widetilde{W}_t)$。

2.6　控制层

经过上一层的交互，在 H 句的各个时刻t，都会输出一个矩阵$W_{\ell_P}^t$。与编码层类似，本层的核心也是一个 GRU 编码器，但它在将交互信息融入到对 H 句编码的同时，也控制着两句间信息的流动程度，因此我们将本层称为控制层。不同于已有强交互模型通过修改编码器计算步骤，将两句的交互信息作为一个输入源接入当前句的编码计算，本文模型不对 GRU 本身做任何修改，而是利用矩阵化带来的优势，使用交互层的输出矩阵动态生成 GRU 计算所需要的三个权重矩阵-更新门W_z^t、重置门W_r^t和候选激励W_h^t。

注意到交互层的输出矩阵（$\sqrt{d}\times\sqrt{d}$）和本层的两个输入x_t、h_{t-1}（d）在维度上并不对称，为此，本文提出两种方式来实现这个拥有动态权重的 GRU 控制层：

a)DIN-1。将两个输入的d维向量转换成$\sqrt{d}\times\sqrt{d}$的矩阵，具体如下：

$$\boldsymbol{H}_{t-1} = \mathrm{TO-MATRIX}(\boldsymbol{h}_{t-1}) \quad (9)$$

$$\boldsymbol{Z}_t = \sigma(\boldsymbol{W}_z^t \cdot [\boldsymbol{H}_{t-1}, \widetilde{\boldsymbol{W}}_t] + \boldsymbol{b}_z^t) \quad (10)$$

$$\boldsymbol{R}_t = \sigma(\boldsymbol{W}_r^t \cdot [\boldsymbol{H}_{t-1}, \widetilde{\boldsymbol{W}}_t] + \boldsymbol{b}_r^t) \quad (11)$$

$$\widetilde{\boldsymbol{H}}_t = \tanh(\boldsymbol{W}_h^t \cdot [\boldsymbol{R}_t * \boldsymbol{H}_{t-1}, \widetilde{\boldsymbol{W}}_t] + \boldsymbol{b}_h^t) \quad (12)$$

$$\boldsymbol{H}_t = (1-\boldsymbol{Z}_t) * \boldsymbol{H}_{t-1} + \boldsymbol{Z}_t * \widetilde{\boldsymbol{H}}_t \quad (13)$$

$$\boldsymbol{h}_t = \mathrm{VECTORIZE}(\boldsymbol{H}_t) \quad (14)$$

其中：

$$\begin{pmatrix}\boldsymbol{W}_z^t\\ \boldsymbol{W}_r^t\\ \boldsymbol{W}_h^t\end{pmatrix} = \tanh\begin{pmatrix}\boldsymbol{W}_{con}^z & \boldsymbol{b}_{con}^z\\ \boldsymbol{W}_{con}^r W_{\ell_p}^t + & \boldsymbol{b}_{con}^r\\ \boldsymbol{W}_{con}^h & \boldsymbol{b}_{con}^h\end{pmatrix} \quad (15)$$

$$\begin{pmatrix}\boldsymbol{b}_z^t\\ \boldsymbol{b}_r^t\\ \boldsymbol{b}_h^t\end{pmatrix} = \tanh\begin{pmatrix}\boldsymbol{B}_{con}^z\\ \boldsymbol{B}_{con}^r \boldsymbol{W}_{\ell_p}^t\\ \boldsymbol{b}_{con}^h\end{pmatrix} \quad (16)$$

$$W_{con}^* \in \mathbb{R}^{\sqrt{d}\times\sqrt{d}}，B_{con}^* \in \mathbb{R}^{\sqrt{d}\times\sqrt{d}}，b_{con}^* \in \mathbb{R}^{\sqrt{d}\times\sqrt{d}}$$

b)DIN-2。将交互层输出的$\sqrt{d}\times\sqrt{d}$的矩阵转换成d维向量，具体如下：

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_z^t \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_{t-1}] + \boldsymbol{b}_z^t) \quad (17)$$

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r^t \cdot [\boldsymbol{h}_{t-1}, \boldsymbol{x}_{t-1}] + \boldsymbol{b}_r^t) \quad (18)$$

$$\widetilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_h^t \cdot [\boldsymbol{r}_t * \boldsymbol{h}, \boldsymbol{x}_{t-1}] + \boldsymbol{b}_h^t) \quad (19)$$

$$\boldsymbol{h}_t = (1-\boldsymbol{z}_t) * \boldsymbol{h}_{t-1} + \boldsymbol{z}_t * \widetilde{\boldsymbol{h}}_t \quad (20)$$

其中：

$$\boldsymbol{V}_{\ell_p}^t = \mathrm{VECTORIZE}(\boldsymbol{W}_{\ell_p}^t) \quad (21)$$

$$\begin{pmatrix}\boldsymbol{W}_z^t\\ \boldsymbol{W}_r^t\\ \boldsymbol{W}_h^t\end{pmatrix} = \begin{pmatrix}\boldsymbol{W}_{con}^z\\ \boldsymbol{W}_{con}^r \boldsymbol{V}_{\ell_p}^t\\ \boldsymbol{W}_{con}^h\end{pmatrix} \quad (22)$$

$$\begin{pmatrix}\boldsymbol{b}_z^t\\ \boldsymbol{b}_r^t\\ \boldsymbol{b}_h^t\end{pmatrix} = \begin{pmatrix}\boldsymbol{B}_{con}^z\\ \boldsymbol{B}_{con}^r \boldsymbol{V}_{\ell_p}^t\\ \boldsymbol{b}_{con}^h\end{pmatrix} \quad (23)$$

$$\boldsymbol{W}_{con}^* \in \mathbb{R}^{d\times d}，\boldsymbol{b}_{con}^* \in \mathbb{R}^{d}$$

TO − MATRIX和VECTORIZE分别指的是矩阵化和向量化。两种方法的区别在于：DIN-1 计算稍复杂，需要多次在向量和矩阵间进行转换;但相较于 DIN-2，它使用的训练参数更少，但也可能因此而无法学得更多的表达。

2.7　输出层

在模型的最后一层，使用平均池化（average pooling）来组合控制层各时刻的输出，得到一个固定长度的向量：

$$\bar{\boldsymbol{h}} = \frac{1}{\ell_H}\sum_{t=1}^{\ell_H} \boldsymbol{h}_t \quad (24)$$

其中:ℓ_H是 H 句的长度。至此，所有的词或短语级别的推理都被融合来决定最后的判断。接着，将$\bar{h}$输入到一个多层感知机（multi-layer perceptron，MLP），它包含两个全连接层（fully-connected layer）和一个 softmax 分类器：

$$\bar{\boldsymbol{h}} = \mathrm{FC}(\bar{\boldsymbol{h}}) \quad (25)$$

$$\Pr(y|\boldsymbol{P}, \boldsymbol{H}) = \mathrm{softmax}(\boldsymbol{W}^{output}\bar{\boldsymbol{h}} + \boldsymbol{b}^{output}) \quad (26)$$

其中:$W^{output} \in \mathbb{R}^{3\times d}$, $b^{output} \in \mathbb{R}^3$。$\Pr(y|P,H) \in \mathbb{R}^3$就是式（1）中的条件概率，最后选择概率值最大的那一维的标签作为预测标签。

2.8　复杂度分析

为了让读者对 DIN 模型的训练效率有更好的理解，本节对模型的复杂度进行一定的分析。为了简化表述，词向量和编码向量的长度被统一定义为d，并且用ℓ来表示句子长度。

在编码层，GRU cell 中权重矩阵（$d\times d$）和向量（d）的乘法运算的复杂度是$O(d^2)$。因此，对整句编码的计算复杂度是$O(\ell d^2)$。在变换层，将两句从原先的词向量（d）表示向矩阵空间（$\sqrt{d}\times\sqrt{d}$）投射的复杂度是$O(\ell d^2)$。在交互层，类似嵌套循环的结构使它的复杂度变为$O(\ell^2 d)$。在控制层，根据动态权重生成方法的不同，相较于复杂度是$O(\ell d)$的 DIN-1，DIN-2 由于采用了重参数化方法（re-parameterization），使复杂度提高到$O(\ell d^2)$。综上，DIN-1 和 DIN-2 的整体复杂度分别为$O(\ell d^2 + \ell^2 d + \ell d)$和$O(\ell d^2 + \ell^2 d + \ell d^2)$。根据观察，$\ell$一般都小于$d$，如在 SNLI 数据集里 P 句、H 句的平均长度和最大长度分别为 34 和 68、28 和 58，而预训练词向量维度一般都在 300 左右。因此，两个模型的复杂度主要都集中在复杂度为$O(\ell^2 d)$的交互层上。

2.9　训练目标

由于 RTE 的本质是一个分类问题，本文采用交叉熵损失（cross entropy loss）作为训练的目标函数。具体地，当给定训练集里各个句子对的真实标签（ground-truth label）y_i和训练参数集θ，目标函数可组织如下：

$$J(\theta) = -\sum_i^N y_i \log y_i^* + \frac{\lambda}{2}||\theta||^2 \quad (27)$$

其中:N是训练集的大小，y_i^*可由式（1）求得。λ是ℓ_2正则化参数，通过[0.0，1E-4，3E-4，1E-3]内小规模的网格搜索（grid search）

后发现，使用ℓ_2正则项反而会拖慢训练过程，还伴有一定程度的性能损失。因此，训练时不采用ℓ_2正则项来强制约束训练参数的更新。

3 实验分析

3.1 实验设置

3.1.1 数据集

实验使用 Bowman 等人[13]于 2015 年发布的 SNLI 数据集，它被广泛用于测试为 RTE 任务设计的神经网络模型的性能。此数据集总共包含 570,152 个句子对，每个句子对都被人工标记了以下标签之一：entailment（蕴涵），contradiction（矛盾），neutral（中立）和-（特殊标签，表示多位标注者未在此句子对上得到一致的标记意见）。和大部分工作的处理方式相同，本文去掉带有-的句子对后按照 549,367/9,842/9,824 的比例划分训练集/验证集/测试集。

3.1.2 模型参数

从训练集/验证集/测试集中一共收集到34 877个不同的词，以此作为训练的词表（vocabulary）。其中，有 30 626 个词能够在预训练词向量集 840B-GloVe[21]中找到。对 OOV（out-of-vocabulary）词，使用[-0.05,0.05]内的随机分布向量作为初始化词向量。词向量的长度固定为 300。编码向量的长度由于要便于开方操作，分别设计了[16*16，18*18，20*20]三组进行实验比较。需要注意的是，虽然同时训练词向量会带来一定的性能提升，但考虑到词向量矩阵十分庞大，所带来的内存负载也是巨大的，因此本实验不会在训练过程中更新词向量矩阵。小批量训练（mini-batch training）的大小为 128，会在一个 batch 中长度不够的句子后面补上额外的 null 标记，它对应的是一个 300 维的 0 向量。训练迭代数（training epochs）为 30，当连续 3 次迭代在验证集上的准确度没有提升甚至出现降低后便提前停止训练（early stopping）。我们会保存下在验证集上准确度最高的那个模型，作为最优模型来对测试集进行预测。

3.1.3 超参数

采用 Kingma 等人[23]提出的 ADAM 作为梯度下降优化器。其中，第一动量系数（first momentum coefficient）β_1和第二动量系数分别设为 0.9 和 0.999。初始学习率（initial learning rate）为 0.001。为了加速梯度下降过程，为全局每 1000 训练步长（training steps）设置了 0.95 的衰减率（decay rate）；同时观察到在 SNLI 数据集上测试的端到端模型特别容易发生过拟合（overfitting）的现象，因此引入 dropout 机制[24]，在编码层的输入和输出端、控制层的输出端随机关闭 20%的神经元。

3.2 定量分析

为了更全面地评估 DIN 的性能，表 2 列出了一系列已有模型进行对比。其中，Para 是除词向量以外模型用到的训练参数的个数，Train 和 Test 分别代表在训练集和测试集上的准确度（%）。

表 2 已有模型和本文模型在 SNLI 数据集上的表现结果对比

编号	模型	Para	Train	Test
1	128D LSTM encoder	1.4M	83.9	81.8
2	100D *word-by-word* attention[5]	250k	85.3	83.5
3	600D DF-LSTM[9]	2.8M	85.9	85.0
4	50D stacked TC-LSTMs[10]	190k	86.7	85.1
5	300D mLSTM[6]	1.9M	92.0	86.1
6	300D LSTMN with deep attention fusion[16]	1.7M	87.3	85.7
7	450D LSTMN with deep attention fusion	3.4M	88.5	86.3
8	200D decomposable attention model[25]	382k	89.5	86.3
9	-with intra-sentence attention	582k	90.5	86.8
10	300D rLSTM[7]	2.0M	90.7	87.5
11	128D HyperLSTM (our implementation)[12]	3.4M	88.1	83.2
12	300D rLSTM (our implementation)[7]	2.0M	88.9	86.6
13	16×16D DIN-1	498k	89.5	86.7
14	16×16D DIN-2	889k	88.9	86.0
15	18×18D DIN-1	718k	90.2	87.4
16	18×18D DIN-2	1.3M	89.6	86.8
17	20×20D DIN-1	1.0M	90.5	**88.0**
18	-without dynamic generated weights	1.9M	89.8	86.2
19	20×20D DIN-2	1.9M	90.1	87.2

同时也实现了两个基准模型 HyperLSTM[12]和 rLSTM[7]，有以下发现：

a)HyperLSTM（11）使用一个基于主 LSTM 的小 LSTM 来动态生成主 LSTM 的权重，它原本为单句建模设计，现置于句子对场景下，相当于要同时训练四个 LSTMs，训练权重数量增长部分拖慢了训练，导致更严重的过拟合现象产生。不仅如此，模型参数设置复杂，状态向量和词嵌入向量的维度间难以平衡。虽然模型表现不理想（83.2%）。但比较 LSTM encoder（1），它的优势仍明显，证明动态权重在 RTE 任务下是有效的。

b)rLSTM（12）作为一种典型的强交互模型，逐词匹配的操作方式和 DIN 类似。虽然尽量沿用了文献[6]给出的实验参数，但由于某些重要参数无法获得，导致测试集准确度（86.6%）与论文给出结果（87.5%）有一定的出入。但相较于 *word-by-word* attention（2），性能提升仍然相当明显，证明强交互性在 RTE 任务下是有效的。

c)本文提出的两种 DIN 模型的性能都随着交互矩阵规模的变大稳步提升。性能最佳出现在 20×20D DIN-1（17），超过已有的最佳模型 300D rLSTM（9），并且用到的训练参数仅为后者的一半。

d)（18）采用削减法（ablation method），去掉 20×20D DIN-1（17）控制层中的权重动态生成操作，将交互层输出矩阵转换成向量后直接作为 GRU 编码器的一个输入。可以看到，仅剩

矩阵交互的 DIN 模型在测试集上的准确度达到 86.2%，虽然低于 rLSTM（12），但也超过了表 2 中大部分的已有模型。另一方面，比完整模型（17）少的 1.8%测试集准确度也进一步证明了动态权重的有效性。

e)有趣的是，尽管 DIN-2 在动态生成权重时用到的训练参数数量超过 DIN-1，它在测试集上的表现却始终差于后者，这与我们在 3.6 节最后的预估相反。可能原因是，DIN-1 不仅在生成动态权重时保持了矩阵计算（式(15)(16)），GRU 的编码步骤也全部矩阵化（式(9)~(14)），保证了和交互层在计算上的连贯性，也从侧面说明矩阵比向量能承载更多的语义信息。

实验结果证明，通过矩阵化和动态权重两者的相互促进，模型能在保持精简结构（训练参数少于大多数已有模型）的前提下，获得更高的测试集识别准确度。

3.3　定性分析

为了更直观地理解 DIN 模型的交互性，通过如下方式获得 H 句词x_t与 P 句的交互向量：

$$V_\tau^t = \text{VECTORIZE}(W_\tau^t) \in \mathbb{R}^d \tag{28}$$

$$\alpha_\tau^t = \ell_2 - \text{norm}\|V_\tau^t\|_2 \tag{29}$$

$$i_\tau^t = \frac{\alpha_\tau^t - \{\alpha_\tau^t\}_{min}}{\{\alpha_\tau^t\}_{max} - \{\alpha_\tau^t\}_{min}} \in [0,1] \tag{30}$$

$$I_t = \{i_\tau^t\} \tag{31}$$

其中：$W_\tau^t, \tau \epsilon [1, \cdots, \ell_P]$是各时刻$t \in [1, \cdots, \ell_H]$在交互层的输出矩阵，而交互向量中的各个元素$i_\tau^t$代表的是当前词$x_t$与 P 句词$x'_\tau$的匹配程度。

图 2 中使用热力图（heat map）来展示各时刻的交互向量，三个例子是手工从测试集里选取的，它们拥有相同的 P 句：a guy in glasses is biting into a pink marshmallow chick while somebody else is puckering their lips out wanting a bite（一个戴眼镜的男孩嘴里吃着小鸡棉花糖，另一个人嘟着嘴也想尝一尝）。第一个例子（图 2 上），模型首先正确识别出了两个近似词对（man，guy）和（eats，biting）。如果仅以此为依据，句子对会被误判为存在蕴涵关系。但之后，模型又挖掘出了 hamburger 和 marshmallow 间明显的矛盾关系，因此句子对逻辑关系反转，最终判定为矛盾关系。第二个例子（图 2 中），虽然 P 和 H 句都有较长的文本结构，但大部分被明确识别的词对都只存在近似关系，如（desires，wanting），因此模型将其判定为蕴涵关系。第三个例子（图 2 下）与前一个例子类似，也有着相当长的文本结构，但在 H 句中多了诸如 his last（他最后一口的）这样在 P 句中不存在修饰句和 his friend（他的朋友）这样在 P 句中不存在的人物关系，还多了额外的一个转折句 but he does not give in（但他并未给他尝）。这些多余信息在 P 句中都无法找到对应面（counterparts），使模型无法对两句关系做出明确判定，因此给出中立的判断。

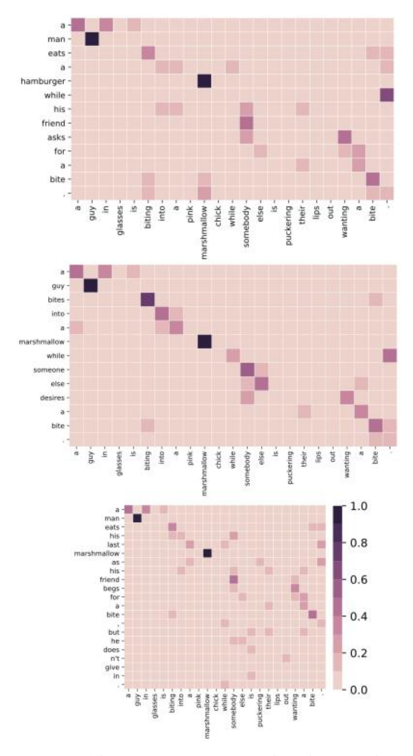


图 2　展示最佳模型 20×20D DIN-1 交互性的三个例子。

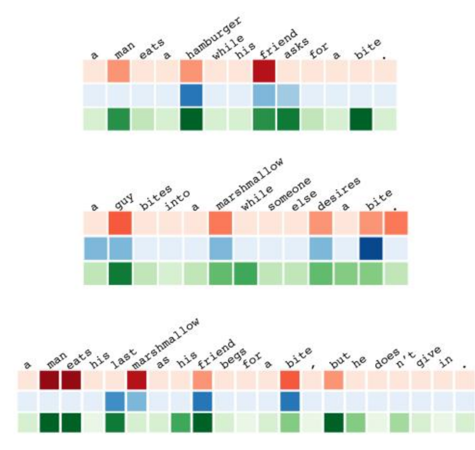


图 3　展示控制层 GRU 编码器权重动态变化的三个例子。

另一方面，也采用同样的方法来展示控制层权重在各时刻的动态变化。图 3 中的三个例子各自对应图 2 中的三个 H 句。红、蓝、绿三色分别代表控制层 GRU cell 的更新门（W_z^t）、重置门（W_r^t）和候选激励权重（W_h^t）在各时刻的变化，变化范围都为[0,1]。有如下两点观察：

a)三个权重色块都相对较浅的词，如 a，while，his（图 3 上），else（图 3 中），as，for，does，in（图 3 下），它们在图 2 中并未产生影响模型判断结果的重要的对齐信息，意味着这些词无法刺激模型对权重进行修改来编码无用信息，此时权重的动态生成处于休眠状态，控制层只着重编码 H 句的上下文信息。

b) 虽然模型在各时刻生成的三个计算权重间此消彼涨过于随机，无固定规律可循，但如果从整体上看三个权重在所有时刻出现的深色块数量，可以发现候选激励权重变化相对频繁，表明候选激励在各时刻的差异性更明显，对两句信息的流动起更主要的控制作用。

4 结束语

为解决识别文本蕴涵问题，本文提出一种动态交互网络 DIN，在 H 句逐词与 P 句匹配的过程中，结合矩阵化和动态网络权重两种方法，将词向量投射到二维矩阵空间进行交互，并利用输出矩阵为 GRU 编码器动态生成计算权重。在 SNLI 数据集上的实验结果表明，本文提出的最优模型超过了原有最佳，且使用的训练参数仅为后者的一半。此外利用削减法，在没有动态网络权重的情况下，单纯凭借矩阵交互，模型的识别准确度也超过了多数交互模型，进一步证明了两种方法各自的有效性。类似 DIN 的强交互模型性能普遍优于弱交互或无交互模型，但却鲜少有相关研究。未来也将继续探索此类模型，一个可行的方向是让交互信息同样回流到原本已固定的 P 句中，促进两句间信息真正的双向流动，从而提升识别准确度。

参考文献：

[1] Huo Huan, Zhang Wei, Liu Liang, *et al.* Collaborative filtering recommendation model based on convolutional denoising auto encoder [C]// Proc of the 12th Chinese Conference on Computer Supported Cooperative Work and Social Computing. Chongqing China: ACM Digital Library, 2017: 64-71. (霍欢，张薇，刘亮，等. 融合卷积降噪自动编码器的协同过滤推荐模型 [C]// 第 12 届全国计算机支持的协同工作与社会计算学术会议论文集. 中国重庆: ACM Digital Library, 2017: 64-71.)

[2] Liang Liu, Huan Huo, Xiufeng Liu, *et al.* Recognizing textual entailment with attentive reading and writing operations [C]// Proc of the 23rd International Conference on Database Systems for Advanced Applications. Gold Coast Australia: Springer, 2018: 1-14.

[3] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, *et al.* Empirical evaluation of gated recurrent neural networks on sequence modeling [EB/OL]. (2014-12-11) [2018-03-15]. https://arxiv. org/abs/1412. 3555.

[4] 霍欢，张薇，刘亮，等. 一种针对句法树的混合神经网络模型 [J]. 中文信息学报, 2017, 31 (06): 58-66. (Huo Huan, Zhang Wei, Liu Liang, *et al.* A hybrid neural network model on syntax tree structures [J]. Journal of Chinese Information Processing, 2017, 31 (06): 58-66.)

[5] Rocktäschel Tim, Grefenstette Edward, Hermann Karl Moritz, *et al.* Reasoning about entailment with neural attention [EB/OL]. (2015-09-22) [2018-03-15]. https://arxiv. org/abs/1509. 06664

[6] Shuohang Wang, Jing Jiang. Learning natural language inference with LSTM [C]// Proc of Conference of the North American Chapter of the Association for Computational Linguistics. Stroudsburg: ACL, 2016: 1442-1451.

[7] Lei Sha, Baobao Chang, Zhifang Sui, *et al.* Reading and Thinking: Re-read LSTM Unit for Textual Entailment Recognition [C]// Proc of International Conference on Computational Linguistics. Stroudsburg: ACL, 2016: 2870-2879.

[8] Biao Liu, Minlie Huang, Song Liu, *et al.* A Sentence Interaction Network for Modeling Dependence between Sentences [C]// Proc of Annual Meeting of the Association for Computational Linguistics. Stroudsburg: ACL, 2016: 558-567.

[9] Pengfei Liu, Xipeng Qiu, Jifan Chen, *et al.* Deep Fusion LSTMs for Text Semantic Matching [C]// Proc of Annual Meeting of the Association for Computational Linguistics. Stroudsburg: ACL, 2016: 1034-1043.

[10] Pengfei Liu, Xipeng Qiu, Yaqian Zhou, *et al.* Modelling Interaction of Sentence Pair with coupled-LSTMs [C]// Proc of Conference on Empirical Methods in Natural Language Processing. Stroudsburg: ACL, 2015: 1703-1712.

[11] Barker C. Continuations in natural language [EB/OL]. (2013-12-10) . http://www. cs. bham. ac. uk/~hxt/cw04/barker. pdf.

[12] Ha David, Dai Andrew, Le Quoc V. Hyper networks [EB/OL]. (2016-09-27) [2018-03-15]. https://arxiv. org/abs/1609. 09106v4.

[13] Samuel R. Bowman, Gabor Angeli, Christopher Potts, *et al.* A large annotated corpus for learning natural language inference [C]// Proc of Conference on Empirical Methods in Natural Language Processing. Stroudsburg: ACL, 2015: 632-642.

[14] Jason Weston, Sumit Chopra, Antoine Bordes. Memory networks [EB/OL]. (2014-10-15) [2018-03-15]. https://arxiv. org/abs/1410. 3916v11.

[15] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, *et al.* End-to-end memory networks [C]// Advances in Neural Information Processing Systems. Cambridge: MIT Press, 2015: 2440-2448.

[16] Cheng Jianpeng, Dong Li, Lapata Mirella. Long short-term memory-networks for machine reading [C]// Proc of Conference on Empirical Methods in Natural Language Processing. Stroudsburg: ACL, 2016: 551-561.

[17] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate [EB/OL]. (2014-09-01) [2018-03-15]. https://arxiv. org/abs/1409. 0473.

[18] Lakoff G. Linguistics and natural logic [J]. Synthese, 1970, 22 (1): 151-271.

[19] MacCartney Bill. Natural language inference [D]. Stanford: Stanford University, 2009.

[20] Milne R, Strachey C. A theory of programming language semantics [M]. London: Chapman and Hall, 1977.

[21] Jeffrey Pennington, Richard Socher, Christopher D. Manning. Glove: Global vectors for word representation [C]// Proc of Conference on Empirical Methods in Natural Language Processing. Stroudsburg: ACL, 2014: 1532-1543.

[22] Tomas Mikolov, Ilya Sutskever, Kai Chen, *et al.* Distributed representations of words and phrases and their compositionality [C]// Advances in Neural Information Processing Systems 26. Cambridge: MIT Press, 2013: 3111-3119.

[23] Diederik P. Kingma, Jimmy Ba. Adam: A method for stochastic optimization [EB/OL]. (2014-12-22) [2018-03-15]. https://arxiv. org/abs/1412. 6980v9.

[24] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, *et al.* Dropout: a

simple way to prevent neural networks from overfitting [J]. Journal of Machine Learning Research, 2014, 15 (1): 1929-1958.

[25] Ankur P. Parikh, Oscar Täckström, Dipanjan Das, *et al.* A decomposable attention model for natural language inference [C]// Proc of Conference on Empirical Methods in Natural Language Processing. Stroudsburg: ACL, 2016: 2249-2255.